

FEB 03 2006

Patent

Attorney Docket No.: Intel 2207/13056

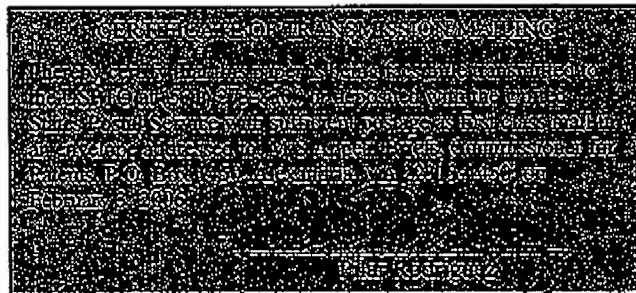
Serial No.: 10/000,335

Assignee: Intel Corporation

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

APPLICANT : Vedvyas SHANBHOGUE  
SERIAL NO. : 10/000,335  
FILED : December 4, 2001  
FOR : ROLLING SOFTWARE UPGRADES FOR FAULT  
TOLERANT SYSTEMS  
GROUP ART UNIT : 2114  
EXAMINER : Timothy M. Bonura  
CUSTOMER NO. : 25693

M/S: APPEAL BRIEFS - PATENTS  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

**APPEAL BRIEF**

Dear Sir:

This brief is in furtherance of the Notice of Appeal, filed in this case on October 3, 2005.

02/06/2006 LWONDIM1 00000066 110600 10000335

01 FC:1402 500.00 DA

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

**1. REAL PARTY IN INTEREST**

The real party in interest in this matter is Intel Corporation. (Recorded December 4, 2001, Reel/Frame 012353/0350).

**2. RELATED APPEALS AND INTERFERENCE**

There are no related appeals.

**3. STATUS OF THE CLAIMS**

Claims 1-18 are pending in the application. Claims 1-18 are rejected under 35 U.S. C. §103 (a) as being unpatentable over Cook, et al., U.S. Patent No. 5,966,304.

**4. STATUS OF AMENDMENTS**

Applicants did not make any amendments to the claim subsequent to final rejection. The claims listed on page 1 of the Appendix attached to this Appeal Brief reflect the present status of the claims (including amendments entered after final rejection).

**5. SUMMARY OF THE CLAIMED SUBJECT MATTER**

The embodiment of claim 1 generally describes a method of upgrading application software on a fault tolerant system having a first engine (e.g., see page 6, line 6, and Figure 2, 60) and a second engine (e.g., see page 6, line 6, and Figure 2, 65), the first and second engine executing an application, the method comprising: taking the second engine out of service (e.g., see page 12, line 19); upgrading the application on the second engine (e.g., see page 12, line 20); assigning the second engine as a standby engine to the first engine and receiving run state

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

updates from the first engine (e.g., see page 12, line 21- page 13, line 7); assigning the first engine as the standby engine to the second engine and receiving run state updates from the second engine (e.g., see page 13, line 7- page 13, line 12); and upgrading the application on the first engine. (e.g., see page 13, line 13- page 13, line 14).

The embodiment of claim 11 generally describes a method for upgrading application software on a fault tolerant system having an active engine and a standby engine, said method comprising: determining if said active engine and said standby engine are executing different versions of said application software (e.g., see page 11, line 2-4); sending a description of work units from the active engine to the standby engine (e.g., see page 13, line 13- page 13, line 14); and sending database activities from the active engine to the standby engine (e.g., see page 11, line 2-4).

The embodiment of claim 15 generally describes a fault tolerant system comprising: a first engine (e.g., see page 6, line 6, and Figure 2, 60); a second engine (e.g., see page 6, line 6, and Figure 2, 65); a computer readable memory that stores instructions that when executed by said first and second engine cause the fault tolerant system to: designate said first engine as an active engine and said second engine as a standby engine (e.g., see page 5, line 15-16); determine if said active engine and said standby engine are executing different versions of an application software (e.g., see page 13, line 7- page 13, line 12); send a description of work units from said active engine to said standby engine (e.g., see page 13, line 13- page 13, line 14); and send database activities from said active engine to said standby engine (e.g., see page 11, line 2-4).

Fig. 1 is a block diagram that illustrates a software application that is to be made fault tolerant according to an embodiment of the present invention.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

Fig. 2 is a block diagram of one embodiment of a fault tolerant system 100 that can be used to implement the present invention. Fault tolerant system 100 includes an active fault tolerance (“FT”) engine 60, a standby FT engine 65, and an application in active/standby redundant pair configuration. The application implements a number of procedures in response to input events that it receives, where each of the procedure is formed by sequencing and grouping a number of work units together.

Fig. 3 illustrates one embodiment of lock-step execution performed by fault tolerant system 100 shown in Fig. 2.

Fig. 4 illustrates a method in accordance with one embodiment of the present invention for performing rolling upgrades of applications using hot standby mechanism for fault tolerance while maintaining lock step synchronization between active and standby applications. In the embodiment shown, each procedure implemented by the application is versioned, and the active and standby copies of the application exchange the procedure version numbers as part of the initialization sequence.

Fig. 5 is a block diagram of a computer system 120 in accordance with one embodiment of the present invention.

Fig. 6 is a flow chart illustrating operations performed in accordance with one embodiment of the present invention for implementing rolling upgrades of computer system 100. In the embodiment described, software is upgraded from “Version 1.0” to “Version 2.0”. In the embodiment described, the operations are stored as software in memories 123, 124 and executed by processors 121, 122. In other embodiments, the operations are performed by any combination of hardware or software.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

At 200, computer system 100 is in an initial state. In this state, processor 121 is an active processor and processor 122 is a standby processor acting as a backup to processor 121. Processors 121, 122 each execute Version 1.0 of the software to be upgraded. Processor 121 keeps processor 122 in synchronization by generating run time state updates such as described in Fig. 4. The rolling upgrade is intended to upgrade system 100 of software Version 2.0.

At 201, processor 122 is isolated by taking it out of service.

At 202, processor 122 is reloaded with Version 2.0 of the software. However, the new features in the Version 2.0 software are disabled. Processor 122 is then integrated into system 100 by being made a standby for processor 121. Processor 121 generates run time state updates to processor 122.

At 203, processor 122 is made the active processor in order to evaluate Version 2.0 of the software. The new features in the Version 2.0 remain disabled. If the software Version 2.0 does not perform acceptably, processor 121 is made the active processor and processor 122 is isolated by taking it out of service, leading to a fallback of Version 1.0 for computer system 100. System 100 is returned to 200.

At 204, if the evaluation was successful, processor 121 is isolated by taking it out of service. System 100 continues to 205.

At 205, processor 121 is reloaded with software Version 2.0. The new features in Version 2.0 of processor 121 are disabled.

At 206, processor 121 is integrated back into system 100 by making it a standby for processor 122. The new features in Version 2.0 continue to be disabled.

At 207, all processors in system 100 have been upgraded, and the new features of software Version 2.0 are then enabled. The rolling upgrade is now completed.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

As described, the rolling upgrade method in accordance with one embodiment of the present invention allows the upgraded version to be validated, yet retains the fault tolerance quality of the software throughout the upgrade process. This allows portable software products to be upgraded on fault tolerant systems.

**6. GROUND'S OF REJECTION TO BE REVIEWED ON APPEAL**

A. Are claims 1-18 anticipated under 35 U.S.C. 102(b) by Cook et al. (U.S. Patent No. 5,966, 304) (hereinafter "Cook")?

**7. ARGUMENT**

A. Claims 1-18 are not anticipated under 35 U.S.C. 102(b) by Cook

Applicants respectfully submit that nowhere in the cited references is the disclosure, teaching or suggestion of at least: "[a] method of upgrading application software on a fault tolerant system ... *assigning the second engine as a standby engine to the first engine and receiving run state updates from the first engine; assigning the first engine as the standby engine to the second engine and receiving run state updates from the second engine ...*" (e.g., as described in the embodiment of claim 1).

Regarding the limitation of "assigning the second engine as a standby engine to the first engine and receiving run state updates from the first engine", the Examiner asserts that column 2, lines 28-32 of Cook discloses a system wherein the secondary controller becomes the main execution controller of the system. Applicants disagree.

First, Applicants respectfully submit that regardless of whether the cited section actually describes a system wherein the secondary controller becomes the main execution controller of

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

the system (Applicants assert it does not), the Examiner fails to cite a section describing receipt of *run state updates* from a first engine as specifically recited in the embodiment of claim 1. Lines 28-32 of column 2 of Cook do not disclose the relevant limitations, as shown below. The cited section states:

Specifically, the present invention provides a primary controller communicating with a secondary controller to, at a switch-over time, cease execution of a user program and cause the secondary industrial controller to begin execution of the user program.

The cited section generically describes a) communication between a primary controller communication with a secondary controller to b) at a certain point stop running a program and c) allow the secondary controller to execute said program. However, nowhere is the teaching suggestion or disclosure of assigning the second engine as a standby engine to the first engine and *receiving run state updates* from the first engine. Indeed, there is no mention of monitoring the run states of either controller at all. In order for Cook to support a proper 102(b) rejection, Cook must disclose each and every limitation of independent claim 1. It does not.

Other sections of Cook confirm this as well. Lines 33-55 of column 9 purport to describe steps to upgrade a functional module according to the embodiments described in the Cook reference. For convenience, the steps in the module are summarized as follows:

1. The module 16a is removed from the rack 14 of processor 12a.
2. Module 16a is upgraded in hardware or firmware while control continues in controller 12b with the controller 12a disqualified.
3. Module 16a is re-installed in controller 12a.
4. Corresponding module 16b to the one upgraded in the controller 12a is removed from controller 12b.
5. Module 16b is upgraded in hardware or firmware while control continues in controller 12a with the controller 12b disqualified.
6. Module 16b is re-installed in controller 12b.

It is apparent that lines 33-55 of column 9 of Cook merely attempts to disclose a method by which two controllers can be upgraded through the use of sequential disqualification.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

However, it is equally clear that nowhere in the cited sections of Cook (*see e.g.*, steps 2 and 5) is the teaching suggestion or disclosure of “[a] method of upgrading application software on a fault tolerant system having a first engine and a second engine, ...the method comprising: ... assigning the first engine as the standby engine to the second engine *and receiving run state updates* from the second engine...” (e.g., as described in claim 1).

Lastly, in the most recent Office Action, the Examiner argues that the Cook discloses “a disqualified controller 12b performs a qualification per process block 116, becoming a qualified secondary controller 12b”, citing lines 60-65 of column 9, and asserts that qualifications constitutes a receiving of state updates, thereby covering the claimed limitations. *See Office Action dated 3/22/2005, paragraph 22n.* Applicants disagree and assert the Examiner’s conclusions are wholly unsupported by the text of Cook. The cited section of Cook, column 9 lines 60-65, states:

Result: Per the flow chart of FIG. 7, insertion is detected at process block 108 and disqualified controller 12b performs a qualification per process block 116, becoming a qualified secondary to controller 12b.

The cited section merely discloses controller 12b performing a qualification process and becoming qualified to secondary controller 12b. The cited section of Cook contains no teaching, suggestion or disclosure of *run state* updates, or any such state updates being *received from the second (or first) controller* (as appropriate). Lastly, contrary to the Examiner’s unsupported assertion, there is no indication *from the Cook reference* that the “qualification” process described in the cited section “constitutes a receiving of state updates”.

Applicants submit the Examiner has failed to show at least the limitations discussed above in the cited prior art. Therefore, since each and every limitation is not taught or suggested in the Cook reference, Applicants submit the Cook reference is inadequate to support proper 35



Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

U.S.C. 102(b) rejections, the rejections should be withdrawn, and independent claim 1 is in condition for allowance. Independent claims 11 and 15 include similar limitations and therefore are also in condition for allowance. Claims 2-10, 12-14 and 16-18 depend from allowable independent claims and therefore are allowable as well.

For at least all the above reasons, the Applicants respectfully submit that this application is in condition for allowance. A Notice of Allowance is earnestly solicited.

Appellants therefore respectfully request that the Board of Patent Appeals and Interferences reverse the Examiner's decision rejecting claims 1-19 and direct the Examiner to pass the case to issue.

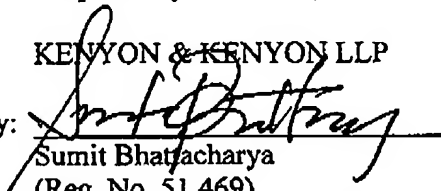
The Examiner is hereby authorized to charge the appeal brief fee of \$500.00 and any additional fees which may be necessary for consideration of this paper to Kenyon & Kenyon Deposit Account No. 11-0600.

Respectfully submitted,

KENYON & KENYON LLP

Date: February 3, 2006

By:

  
Sumit Bhattacharya  
(Reg. No. 51,469)

KENYON & KENYON LLP  
333 West San Carlos St., Suite 600  
San Jose, CA 95110  
Telephone: (408) 975-7500  
Facsimile: (408) 975-7501

82021.1

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

## APPENDIX

(Brief of Appellant Vedvyas SHANBHOGUE  
U.S. Patent Application Serial No. 10/000,335)

### 8. CLAIMS ON APPEAL

1. (Original) A method of upgrading application software on a fault tolerant system having a first engine and a second engine, the first and second engine executing an application, the method comprising:
  - taking the second engine out of service;
  - upgrading the application on the second engine;
  - assigning the second engine as a standby engine to the first engine and receiving run state updates from the first engine;
  - assigning the first engine as the standby engine to the second engine and receiving run state updates from the second engine; and
  - upgrading the application on the first engine.
2. (Original) The method of claim 1, wherein said upgrading the application on the second engine comprises disabling new features of the upgraded application.
3. (Original) The method of claim 1, wherein said upgrading the application on the first engine comprises disabling new features of the upgraded application.
4. (Original) The method of claim 1, further comprising enabling new application features on the first and second engine.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

5. (Original) The method of claim 1, wherein said assigning the first engine as a standby to the second engine comprises determining if the upgraded software is acceptable.

6. (Original) The method of claim 5, wherein if said upgraded software is not acceptable, assigning the first engine as an active engine.

7. (Original) The method of claim 5, wherein if said upgraded software is acceptable, taking the first engine out of service.

8. (Original) The method of claim 1, wherein the application comprises at least one work unit.

9. (Original) The method of claim 8, wherein the run state updates comprise a description of the at least one work unit.

10. (Original) The method of claim 8, wherein said first engine and second engine operate in synchronization.

11. (Original) A method for upgrading application software on a fault tolerant system having an active engine and a standby engine, said method comprising:

determining if said active engine and said standby engine are executing different versions of said application software;

sending a description of work units from the active engine to the standby engine; and

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

sending database activities from the active engine to the standby engine.

12. (Original) The method of claim 11, further comprising:

registering the work units at the standby engine.

13. (Original) The method of claim 11, further comprising:

sending a step-up signal from the active engine to the standby engine with the description of work units.

14. (Original) The method of claim 11, wherein the application software comprises the work units.

15. (Original) A fault tolerant system comprising:

a first engine;

a second engine;

a computer readable memory that stores instructions that when executed by said first and second engine cause the fault tolerant system to:

designate said first engine as an active engine and said second engine as a standby engine;

determine if said active engine and said standby engine are executing different versions of an application software;

send a description of work units from said active engine to said standby engine;

and

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

send database activities from said active engine to said standby engine.

16. (Original) The system of claim 15, said instructions further causing said first and second engine to:

register the work units at the standby engine.

17. (Original) The system of claim 15, said instructions further causing said first and second engine to:

send a step-up signal from the active engine to the standby engine with the description of work units.

18. (Original) The system of claim 15, wherein the application software comprises the work units.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

### **9. EVIDENCE APPENDIX**

No further evidence has been submitted with this Appeal Brief.

Serial No. 10/000,335

Appeal Brief Under 37 CFR 41.37 Filed February 3, 2006

Advisory Action dated October 3, 2005

#### **10. RELATED PROCEEDINGS APPENDIX**

Per Section 2 above, there are no related proceedings to the present Appeal.